IN THE SPECIFICATION:

Please replace the paragraph on page 1, line 7 to page 2, line 4 with the following paragraph:

The client/server environment typically consists of both server and client-side services providing a distributed computing model in which client applications request services from server applications. An example of a client/server system is a business application that allows a user to access information on a central database server. Another example of client/server environment is a web environment that includes a continuously growing number of different applications for web environment support, including web browsers, web servers, launchers, etc. A variety of web resources such as Hyper Text Markup Language (HTML) files, Java Servlet Pages (JSP™), Java™ classes and Servlets™, Enterprise Java Beans™ (EJB), Extensible Markup Language (XML) deployment descriptors makes the task of application development and debugging more complicated and time consuming, especially if more than one developer is involved in a project. An integrated development environment (IDE) is usually used to assist in developing relatively large and complex software products for a client server environment. A developer is usually responsible for developing a piece or module of the final software application. Such a module must be written and debugged prior to its implementation into the final product for seamless integration with other modules. Another problem related to testing of the final product is based on the necessity of using a number of client server environment support applications such as web browsers, application servers, etc. Variations between particular client server environment support applications such as the differences between two application servers from different sources makes the task of a developer more complex, because the developer, in addition to engineering compatibility with other modules of the business applicant, has to provide compatibility with a number of different servers and web browsers. Testing the compatibility of one or more development resources with one or more

servers or one or more client side applications can consume a great amount of the developer's time which extends the duration of product development and makes the product more expensive.

Please replace the three paragraphs on page 8, line 24 to page 9, line 22 with the following paragraphs:

[0034] IDE ~~306~~ 308 is an application for generating code, such as a business application, for running in a client server environment. IDE ~~306~~ 308 includes various editors, tools and wizards for, among other things, creating code, deploying code to a server and testing such code. IDE ~~306~~ 308 is adapted to provide a user interface such as a graphical user interface (GUI) for such operations as is well known to persons of ordinary skill in the art.

[0035] EM ~~308~~ 306 using the EM lists 310, provides a mechanism for executing server side code on one of one or more servers and in association with one of one or more client applications (i.e. "clients"). In accordance with a model of EM ~~308~~ 306, the execution of server side code is partitioned into three stages, the view stage for determining the code for running, the server stage for determining the server to execute the code and the launcher stage to determine the client for interacting with the server to run the code. EM lists 310 extensibly configure EM ~~308~~ 306 to work with different types of server side code to be run (per view list 310a), different servers to run the code (per server list 310b) and different clients with which to interact with the server running the code (per launcher list 310c). The EM lists 310 each comprise one or more respective elements. Each element provides an interface for receiving an input to the element and provides for the processing of the input to produce an output for processing by elements of subsequent stages in the EM model as described further below.

[0036] By modeling the execution of server side code in three stages EM ~~308~~ 306 enables the use of different servers and launchers for executing different code types and provides a mechanism to easily add additional or modify exiting servers, launchers and code types to EM ~~308~~ 306. As described further herein below, each of the EM lists 310a, 310b, 310c of respective input object, server and client elements may be maintained independently of one another and are extensible. For example, a server element for enabling execution of code on a particular server may be added to server list 310b or an existing element in such list amended without regard to the elements of lists 310a and 310c.

Please replace the paragraph on page 10, lines 1 - 8 with the following paragraph:

[0038] FIG. 4 illustrates a schematic diagram of an exemplary structure of a client-server environment and EM ~~308~~ 306 integrated in IDE ~~306~~ 308. A well known client-server environment is a web environment based on a server side part 402 and a client side part 404 shown schematically separated by a dotted line. The illustrated client-server environment for exemplary purposes is embodied using one computing system 100, but it must be understood by those of ordinary skill in the art that the environment may be distributed among one or more computing devices which are enabled to interact with each other using a network such as network 102.

Please replace the two paragraphs on page 10, line 24 to page 11, line 24 with the following paragraphs:

0040] FIG. 5 illustrates a schematic block diagram 500 of EM ~~308~~ 306. EM ~~308~~ 306 comprises a view module 502, a server module 504 and a launcher module 506. Each of the modules 502, 504 and 506 may be implemented as a reusable piece of computer code with specific inputs and outputs. The modules 502, 504 and 506 are preferably adapted to be plugged in and/or removed easily from an application, for example, IDE ~~306~~ 308. Each of the modules 502, 504 and 506 is adapted by a corresponding EM list 310a, 301b and 310c to handle respective inputs and produce respective outputs for handling by subsequent modules in accordance with the EM model described above.

[0041] In order to describe operations and features of EM ~~306~~ 308 and system 100, the following definitions are used. A deployable object is an output, such as a Java code object, XML document etc., that contains information about code to be run on an as yet unidentified server. Depending on the type of code to be run, the deployable object therefor contains enough information to publish the code to a server (though not necessarily a specific server) and may contain information on what the resource is "contained" in, (i.e. how it is packaged). For example, a deployable object for an HTTP file may contain a filename or other identifier to construct a Universal Resource Locator (URL) for the HTML file to be run on an HTTP server. A launchable object is an output, such as a Java code object, XML document etc., that contains information on how to access code to be run (i.e. a specific resource) on a particular server, and information about how to route access to the resource, (e.g. firewalls and/or the type of client application that can be used to access the web resource) if the access information is not sufficiently clear. For example, a URL may be constructed and provided to direct access to the resource when a Web browser is intended to consume the object. However, a specific resource may require a specific plug-in in a browser that supports the specific resource. For example, an HTML file may have a
CA920020055US1

Macromedia Shockwave.TM. reference. In order to fully display the HTML page, a browser intended to display the page requires an appropriate Macromedia Shockwave.TM. plug-in installed. The launchable object for such an HTML page contains information specifying that the HTML file requires a browser having Macromedia Shockwave.TM. plug-in installed. Also, a launchable object may contain information such as a URL and an IP address and/or port number.

Please replace the paragraph on page 14, line 27 to page 15, line 12 with the following paragraph:

[0052] As a result of the extensible nature of the EM model, an element may be added to one of the EM lists 310 for which there may not be a suitable element in one or more of the other lists. For example, a new server element may be added to server list 310b for which no suitable client element yet exists in launcher list 310c. In order to support enhanced service for a user and avoid a situation wherein a choice is presented at one stage and then on the next stage the ~~use~~ user is notified that the choice cannot be completed ~~competed~~, in one embodiment of the invention, EM 308 is configured to pre-check the compatibility of inputs (e.g. input object 508, deployable object 511 and launchable object 512) with each corresponding module (e.g. view module 502, server module 504 and launcher module 506) prior the actual processing of input object and displaying dialogs to the user. The system programmatically traces through possible outcomes of the steps to ensure that each step will be allowed to go all the way through. If a given input cannot go through at least each of one of the view module, server module or launcher module, the system does not process the input and will notify the user that the selection cannot be processed.